Review: for and function

range(n)

```
>>> list(range(5))
[0, 1, 2, 3, 4]
>>>
```

- range(n)基本上表達一個由0到n-1的數。
  - range(5)代表0, 1, 2, 3, 4五個整數。
- list(range(n))將range(n)的資料型態轉成list的資料型態，return value是一個list。
  - list(range(5))的return value是:
    [0, 1, 2, 3, 4]

for loop

- for 是一個迴圈，會重複執行相同的東西。
- 在重複執行時，有一個變數會一直在改變。
- 語法:

```
for i in [1,2,3]:
        for block here
next statement
```

```
for i in [1,2,3]:
    print(i)
print(4)
```

```
1
2
3
4
>>>
```

range(m, n) m<n

- range(m, n)基本上表達一個由m到n-1的數。

range(3,6)代表3, 4, 5三個整數。

list(range(3, 6))的return value是:
[3, 4, 5]

```
>>> list(range(3, 6))
[3, 4, 5]
>>>
```

List

- Python中，list是使用中[和]將資料包再一起，中間以逗號','將資料分開。
- [1, 2, 3, 4, 5]代表長度為5的列，[]則代表長度為0的列，和空集合、empty string ''的概念相同，一個甚麼都沒有的列。
- 列裡面可以放不同資料型態的資料，如['Anna', 'Lee', 25, 'female']

Index (lists are mutable)

- List的元素是可以被改變的(mutable)，改變之前必須要能夠取得元素。
- list使用index來獲取元素，list L的第一個元素從0開始，使用L[0]來取得。
- 第二個元素則使用L[1]取得，餘此類推。
- 有五個元素的list，使用L[0]到L[4]來取得這五個元素，L[5]就會有error message。

Operation: +(concatenation)

- +: 連結兩個list，回傳值是將第二個list連接到第一個list的後面。
- list * n會將list中的資料重複n遍
- len(L)回傳L的長度。

```
>>> L1=[1,3,5,7,9]
>>> L2=[2,4,6,8,10]
>>> L1+L2
[1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
>>> [0,1]*2
[0, 1, 0, 1]
>>>
>>> L1=[1,2,3]
>>> L2=[4,5,6,7]
>>> len(L1)
3
>>> len(L2)
4
```

Index (lists are mutable)

```
>>> L=[1,2,3,4,5]
>>> L[0]
1
>>> L[3]
4
>>> L[5]
Traceback (most recent call last):
  File "<pyshell#26>", line 1, in <module>
    L[5]
IndexError: list index out of range
>>>
```

```
>>> L=[1,2,3,4,5]
>>> L[3]
4
>>> L[3]=6
>>> L
[1, 2, 3, 6, 5]
>>> L[0]=L[1]+L[2]
>>> L
[5, 2, 3, 6, 5]
```

lists of list

- list的內部也可以是list。

```
>>> L=[1,2,3,4,5]
>>> L[2]=[6,7,8]
>>> L
[1, 2, [6, 7, 8], 4, 5]
>>> len(L)
5
>>> len(L[2])
3
>>> len(L[3])
Traceback (most recent call last):
  File "<pyshell#89>", line 1, in <module>
    len(L[3])
TypeError: object of type 'int' has no len()
>>>
```

Function 的功用

- 將一個program細分為許多的函數所組成，可以:
  - 簡化程式
  - 容易了解
  - 程式碼可被重複使用
  - 方便測試
  - 易於群體工作與溝通
  - ……
- 分兩種function，value returning function 和 void function。

List in for loop

```
>>> for i in [2,4,6,8,10]:
        print(i)


2
4
6
8
10
>>>
```

```
sum = 0
L=[2,4,6,8,10]
for i in L:
    sum = sum + i
print('The sum for the elements in list', L, 'is', sum)
The sum for the elements in list [2, 4, 6, 8, 10] is 30
>>>
```

Value Returning Function

- Function 包括了:
  - Name(名稱)一個
  - Parameter(參數)0或多個
  - Function Body
  - Return(回傳)一個值
- The int, str, bool are functions.

```
def name(para):
    function body
    (with return value)
```

```
def maxof2(a,b):
    if a>b:
        return a
    else:
        return b
```

Void function

- Void Function 通常幫您做一些事，他的回傳值沒有意義：
- 可以有0個或多個參數

```
def Hello():
    print("Hello!")
def main():
    Hello()
    Hello()
    Hello()
main()
```

```
Hello
Hello
Hello
>>>
```

```
def Hello(title, name):
    print('Dear ', title, name, ':')
    print('Nice to meet you\n')
def main():
    Hello('Mr.', 'Hou')
    Hello('Teacher', 'Chuang')
    Hello('Prof. ', 'Zhou')
main()
```

```
>>>
Dear Mr. Hou :
Nice to meet you

Dear Teacher Chuang :
Nice to meet you

Dear Prof. Zhou :
Nice to meet you
```

Function call: Passing argument to Function

Function call的執行順序:

1. evaluate所有argument的值。
2. 建立一個新的變數與值的表格，依照順序將parameter變數和argment值綁在一起。
3. 執行function body的程式碼，直到碰到return statement為止。
4. Evaluate return statement的值，刪除function call時我建立的變數與值的表格，並將回傳值回傳給呼叫程式。
5. 繼續執行呼叫前的程式。

Function call

- Function call是呼叫函數的方法
- 當function definition裡有需要傳送值進來的參數時，Function call必須提供一些值，這些值稱為argument。

Function Call — argument

```
c = maxof2(3+7, 6+6)
print(c)
```

Function Definition — parameter

```
def maxof2(a,b):
    if a>b:
        return a
    else:
        return b
```

Main()

通常我們需要一個主程式來使用函數:

```
def maxof2(a,b):
    if a>b:
        return a
    return b
def main():
    a = int(input('Input a: '))
    b = int(input('Input b: '))
    print('The larger between', a, 'and', b, 'is', maxof2(a,b))
```